7. Gary

Program Name: Gary.java   Input File: gary.dat

Gary has just encountered boolean postfix expressions in class, and is a bit confused. He understands the easy ones, like, "true and false" becomes "true false and", and knows how to evaluate them according the following rules of thumb:

• “and”: evaluates to “false" only when both operands are “true”
• “or”: evaluates “false" only when both operands are “false”
• “xor”: evaluates to “true" only when the operand values are opposite
• “not”: reverses the value from “true” to “false”, or vice versa

It's evaluating the more complex ones in postfix form that give him a bit of trouble.

For example, "true or false and true" is confusing to him. *"What is the postfix version?"*, he wonders, *"and then what is the answer?"* Well, his teacher helps him out and just gives him the postfix version, for now, which is "true false true and or", and then shows him how to process it.

His teacher says, *"Take the first two boolean values and see if there is an operator immediately after them. If there is, do that operation, and replace the result in place of that expression. If not, ignore the first operand for now, and look at the next pair to see if there is an operator following, and evaluate it and replace it with an answer."*

"In the example, "true false true and or", there is no operator for the first two operands, so you ignore the first operand and look at the next two, which have an "and" after them. Well, "false and true" evaluates to "false", so you replace the "false true and" with just "false", which now gives you the expression, "true false or", which is "true", and is the final value of the expression.

"If you have a "not" in the expression, like this one, "true false or not", the result here would first be, "true not", since "true false or" becomes "true", and then "false", since the "not" operator only needs one operand, and "not true" means "false"."

Input: Several lines of data, each line with a postfix boolean expression made up of the words, "true", "false", "not", "and", "or", and "xor", with no parentheses, all separated by single spaces.

Output: The final **true** or **false** value of the postfix boolean expression.

Sample input:
```
true true or
true false and
true false true and or
true false or not
```

Sample output:
```
true
false
true
false
```